

Aritmetikai operátorok

Matematikából ezeket műveleti jelek néven ismerjük.

- $+$, $-$, $*$, $/$, zárójelek
- Van még a $\%$, ezt nem ismerjük: két egész szám osztásának maradékát adja meg: $5 \% 2$ eredménye 1.
- Logikus lenne, hogy a hatványozás a $^$ jel legyen. Javában nem az (hanem egy olyan operátor, ami a szám bitjeivel végez el egy műveletet (kizáró vagy – nem kell tudni)).
- Lényegük, hogy két szám között végeznek műveletet, eredményük szintén egy szám.
- Ha egy vagy több operátorból és literálisból vagy változóból összepakolunk egy sort, azt *kifejezésnek (expression)* nevezzük. Minden kifejezésnek van értéke és típusa.

Milyen szám lesz az eredmény?

- ha a két szám egyforma típusú (pl. mindkettő `int`), akkor az eredmény is azonos típusú lesz (tehát `int`).
- ha a két szám különböző, akkor a művelet elvégzése előtt a kisebb (`byte`, `short`, `int`, `long`, `float`, `double` sorrendben növekednek) értéket a bővebb típusára konvertálja, automatikusan (ezt hívjuk bővítésnek, programozósan *automatikus* vagy *implicit típuskonverzió*nak, *automatic or implicit type conversion*, *type cast*).
 - szorgalmiként megnézheted, hogy bármelyik számot is veszed a kisebb típusból (pl. `int`), az beletehető a nagyobbba (pl. `double`).
- Pl. $2 + 3.0 \rightarrow 2.0 + 3.0$ (a 2 egész literált „gondolatban” `2.0 double` literállá konvertálja) $\rightarrow 5.0$ (az eredmény `double`).

Példák

- $2 + 3$ – eredménye 5. Mivel a 2 is és a 3 is `int`, ezért az eredmény is `int`.
- $6 * 7$ – eredménye 42. Az előbbiek értelmében ő is `int`.
- $2 + 6 * 7$ – itt először elvégezzük a szorzást (általános iskolából talán még emlékszünk, hogy először a szorzást kell elvégezni), majd az összeadást. Az eredménye: 44. Mivel `int` értékek vettek részt a műveletben, az eredmény típusa is `int`.
- $(2 + 6) * 7$ – eredmény 56 - előbb a zárójelben lévő végézzük el, majd a szorzást.
- $2 + 3.0$ – eredmény 5.0, mivel az egyik típusa `int`, a másiké `double`, az eredmény a `double` (mert a `double` szélesebb az `int`-nél).
- $5.1 * ((2 + 3) / 2.2)$ - eredmény kb. 11,590909... - nincs „kapcsos-” és „szögletes” zárójel, csak „kerek” zárójel van. Először a $2+3$ -at végzi el, a részeredmény 5, ami `int`. Ezt elosztja 2.2-vel, így előbb az 5 egészéből 5.0 tört lesz, az osztás után pedig 2.2727..., ami már `double`. Ezt utána szorozzuk 5.1-gyel, és így kapjuk

meg a végeredményt.

Probléma az osztással

Különbség van közöttük, ha két egész számot osztunk el egymással, vagy ha két törtet.

- $7.0 / 3.0$ – két `double` literál, eredménye az, amit a matekban megszoktunk: $2,33333\dots$ (`double`-ként)
- $7 / 3$ – két egész literál, eredménye 2, azaz a matematikai eredmény, **lefelé kerekítve**. (`int`-ként)
- Az osztási maradékot is megkaphatjuk: $7 \% 3$ eredménye 1.

Ugyanez a jelenség fennáll, ha nem literálok, hanem változók szerepelnek egyik vagy másik oldalon.

```
int a = 7;
int b = 3;
int c = a / b; // c értéke 2 lesz!
```

Ilyenkor persze kézenfekvő, de mi van ezzel a kifejezéssel?

```
3 / 4 * 10.1
```

Zsebszámolóval $7,575$ jön ki. Javában pedig 0 lesz az eredmény, hiszen $3 / 4$ az 0!

Ezt a dolgot a programozást tanulók túlnyomó többsége legalább egyszer elszúrja, még akkor is, ha én a fejemen pörgök, úgy próbálom átadni az infót... ☺ Hátha Te majd odafigyelsz... ☺

Az értékadás értéke

A C stílusú nyelvekben, amilyen a Java is, az `=` is egy operátor.

Ez azt jelenti, hogy ennek a műveletnek is van értéke. Míg a $2 + 2$ kifejezés értéke a 4, addig pl. az `a = 5` értéke az 5. És **mellesleg** az `a` változó felveszi az 5 értéket.

Az értékadás művelet (operátor) értéke mindig az értékül adott érték. Ha `c = 2 + 4` a kifejezés, akkor előbb végrehajtódik a $2 + 4$, ami 6, majd ez értékül adódik a `c` változónak, és a kifejezés értéke is 6 lesz, amit kiírhatunk pl. a képernyőre, vagy egy harmadik változónak értékül adhatunk: `System.out.println(c = 3);` a képernyőre a 3 értéket fogja nyomtatni.

Ebből következik, hogy az értékadást is lehet *láncolni*:

```
a = b = 6;
```

Először végrehajtódik a $b = 6$, aminek az értéke 6, a következő lépésben az $a = 6$ hajtódik végre.

Kérdések

- **A fenti szabály (Probléma az osztással) általánosan vonatkozik az egész és a valós típusú változók mindegyikére? Tehát a long, short, byte viselkedése mint az int $\rightarrow 7 / 3 = 2$ és a float eredménye 2.333333?**
Így van, jól érted. Megj. a kisebb egész típusú változókból `int`-et készít általában a művelet végrehajtásakor.